(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2002－278707
（P2002－278707A）
(43)公開日　平成14年９月27日(2002.9.27)

(51)Int.Cl.[7]	識別記号	ＦＩ		テーマコード（参考）
Ｇ０６Ｆ　3/06	３０５	Ｇ０６Ｆ　3/06	３０５Ｚ	５Ｂ００１
11/10	３２０	11/10	３２０Ａ	５Ｂ０６５

審査請求　未請求　請求項の数１　ＯＬ　（全 16 頁）

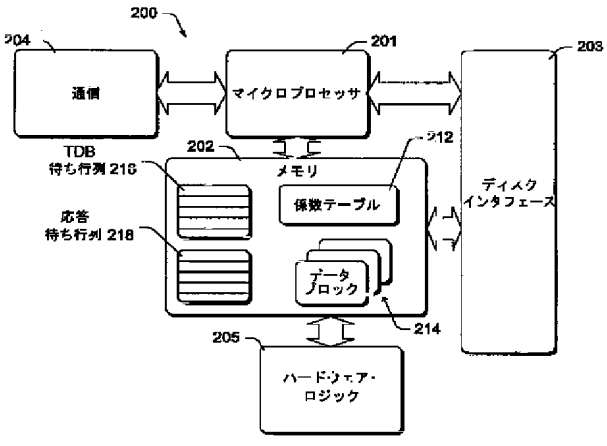
(21)出願番号　特願2002－9660(P2002－9660)
(22)出願日　平成14年１月18日(2002.1.18)
(31)優先権主張番号　０９／８０８，７１３
(32)優先日　平成13年３月14日(2001.3.14)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　398038580
ヒューレット・パッカード・カンパニー
ＨＥＷＬＥＴＴ－ＰＡＣＫＡＲＤ　ＣＯＭＰＡＮＹ
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(72)発明者　バリー・ジェイ・オールドフィールド
アメリカ合衆国83616アイダホ州ボイジー、ウェスト・ダニエル・コート　11302
(74)代理人　100081721
弁理士　岡田　次生　（外２名）

最終頁に続く

(54)【発明の名称】　ディスクコントローラ

(57)【要約】
【課題】　大容量データ格納システムにおいて、タスク記述ブロックを用いて演算に関する情報を維持し、適切なデータスループットを確保する。
【解決手段】　ディスクコントローラはマイクロプロセッサ及び演算ロジックの両方がアクセス可能なメモリを含む。演算を実行するために演算ロジックが必要とする情報は、マイクロプロセッサにより、メモリ内のタスク記述ブロックに格納される。タスク記述ブロックへのポインタは、タスク記述ブロック待ち行列に加えられる。すると、演算ロジックは、待ち行列内のポインタに基づいてタスク記述ブロックにアクセス可能となり、対応する演算を実行する。



【特許請求の範囲】
【請求項１】　冗長データ記憶システムにおいてパリティ演算を実行するディスクコントローラであって、
実行すべき前記パリティ演算についての情報をそれぞれ識別する複数のタスク記述ブロックを格納するメモリと、
前記メモリに接続され、前記複数のタスク記述ブロックのそれぞれにアクセスし、前記タスク記述ブロックで識別される情報に基づいて前記パリティ演算を実行するパリティ演算ロジックと、
を備えるディスクコントローラ。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、実行する演算に関連する情報の維持に関し、特にタスク記述ブロックを使用して演算に関連する情報を維持することに関する。
【０００２】
【従来の技術】最近の大容量データ格納システムは、データの冗長記憶のために複数の物理ディスクドライブを利用することが多い。このような構成により、データアクセス速度が向上すると共に、任意の単一ディスクの障害により発生しうるデータ損失の保護が容易になる。
【０００３】冗長データの記憶には２つの一般的な方法がある。第１の方法、すなわち「ミラーリング」法では、データは複製されて記憶システムの２つの別個の領域に格納される。ディスクアレイには、例えば、同一のデータが２つの別個のディスクに格納される。この方法は、性能が高くまたデータ可用性も高いという利点がある。しかし、ミラーリング法は比較的高価でもあり、データ格納のコストが事実上２倍になる。
【０００４】第２の方法、すなわち「パリティ」法では、記憶領域の一部を使用して冗長データを格納するが、冗長記憶領域のサイズは、元のデータの格納に使用される残りの記憶空間よりも小さい。例えば、６枚のディスクを有するディスクアレイでは、５枚のディスクをデータの格納に使用し、６枚目のディスクを「パリティ」データと呼ばれる冗長データの格納専用とする。パリティデータを生存ディスクからのデータと併せて使用することで、１枚のデータディスクからデータを再構築できるようになる。パリティ法は、ミラーリング法よりも費用が低いため有利であるが、同時にミラーリング法と比較して性能が低く可用性も低いという特徴がある。
【０００５】パリティ記憶を利用する従来のディスクアレイでは、記憶ディスクにおける空間は、複数の記憶ディスクにわたる複数の記憶ストライプに構成される。各ストライプは複数の記憶空間セグメントからなり、各セグメントは、ディスクアレイの単一記憶ディスク上にあるストライプの一部分である。
【０００６】図１は、６つの記憶ディスク13を有する従来のディスクアレイ12を示す。この単純化した例では、複数の記憶ディスクにわたる５つの記憶ストライプがある。図１は、これら５つのストライプのうちの１つのストライプのデータセグメント及び記憶セグメントを強調して示している。強調して示されたストライプのデータセグメントは、網掛けで示されている。このストライプに対応するパリティセグメントは、黒塗りで示されている。一般に、任意の所与のストライプを構成する６つのセグメントのうち、５つのセグメントはデータセグメントであり、６番目のセグメントがパリティセグメントである。
【０００７】この種のパリティ記憶は５＋１パリティ記憶と呼ばれ、単一のパリティセグメントにつき５つのデータセグメントがあることを示している。この方式はより一般的にＮ＋１グループ化と呼ばれる。ここで、Ｎはデータストライプにおけるデータセグメントの実際の数である。
【０００８】図１に示すようなＮ＋１冗長グループ化は、任意の１つの物理記憶装置をデータ損失から保護する。記憶装置に障害が発生した場合、生存データからそのデータを再構築することができる。データを回復するために実行される計算は単純であり、この計算は既知である。一般に、次式に従って、データセグメントD_0～D_{N-1}から単一のパリティセグメントＰが計算される。
【０００９】
【数１】

$$P = x_0 + x_1 + x_2 + x_{N-1}$$

式中、x_0～x_{N-1}は、データセグメントD_0～D_{N-1}からのデータに相当する。任意の単一データセグメントを損失した後、同じ式を簡単に変形することでそのデータを回復することができる。
【００１０】しかし、多くのシステムでは、２つ以上の記憶装置を損失から保護することが重要になりつつある。そのため、冗長格納システムにおいてＮ＋２グループ化を実施することが必要になりつつある。
【００１１】Ｎ＋２冗長グループ化によりデータ保護が強化されるが、その一方、最初のパリティセグメントの計算及び任意の損失データセグメントの再構築の両方で、より複雑な計算が必要となる。
【００１２】Ｎ＋２パリティ計算の一般式は、次のようになる。
【００１３】
【数２】

$$P = p_0x_0 + p_1x_1 + p_2x_2 + p_{N-1}x_{N-1}$$

$$Q = q_0x_0 + q_1x_1 + q_2x_2 + q_{N-1}x_{N-1}$$

式中、Ｐは第１のパリティセグメントの値であり、Ｑは第２のパリティセグメントの値であり、x_0～x_{N-1}はデータセグメントの値であり、p_0～p_{N-1}及びq_0～q_{N-1}は所与のパリティ方式に固有の定数係数で

ある。

【００１４】これらの式は、線形代数の規則により、任意の２つの障害が発生した記憶装置の単一ストライプからのデータを表す任意の２つの未知のx_a～x_bについて潜在的に解くことができる２つの等式システムを形成する。１つの要件は、２つの係数セットp_i及びq_iが一次独立していることである。この要件は、例えば、$p_0=1$、$p_1=1$、$p_2=1$、かつ$q_0=1$、$q_1=2$、$q_2=3$の場合に満たされる。また他の例も可能である。

【００１５】Ｎ＋２パリティの数学的処理は既知であり、本説明の主題ではない。しかし、上記の簡潔な説明から、Ｎ＋２パリティの計算はＮ＋１パリティの計算よりもかなり複雑であることは明白である。Ｎ＋２ディスクアレイの実際の実施において、この複雑さは、記憶装置コントローラ及び結果的にディスクアレイ全体のデータスループットを制限する恐れがある。

【００１６】

【発明が解決しようとする課題】発明者Michael B. Jacobsonによる「Efficient Parity Operations（効率的なパリティ演算）」という名称の同時係属中の米国特許出願（代理人整理番号10971442）には、Ｎ＋２パリティ計算により複雑性が追加されるにも関わらず、適切なデータスループットを維持するより効率的な方法が記載されており、この出願は参照により本明細書に援用される。しかし、この同時係属中の出願に記載のより効率的な方法及び手段を用いても、記憶装置コントローラのデータスループットは、パリティ計算の実行に必要な情報を格納するためのメモリ構造が非効率的であることによりなお悪影響を受けることがある。本発明は、タスク記述ブロックを用いて、パリティ（または他の）演算に関する情報を維持し、適切なデータスループットを維持する方法及び装置を提供する。

【００１７】

【課題を解決するための手段】本明細書では、タスク記述ブロックを用いた演算に関連する情報の維持を説明する。

【００１８】本発明の一態様によると、タスク記述ブロック待ち行列は複数のタスク記述ブロックそれぞれを指すポインタを格納し、各タスク記述ブロックは１つの演算についての情報を格納する。演算のデータはメモリに格納される。各タスク記述ブロックは、メモリに格納されているデータを指すポインタと、演算ロジックが演算の結果を格納すべきメモリ内の結果位置を指すポインタとを含めた、演算ロジックが単一の演算を実行するために必要な情報を格納する。この情報がすべてタスク記述ブロックに格納されると、タスク記述ブロックを指すポインタがタスク記述ブロック待ち行列に格納され、演算ロジックが要求するときにいつでもその演算を実行することが可能になる。

【００１９】パリティ演算に関連する本発明の別の態様によると、タスク記述ブロック待ち行列は複数のタスク記述ブロックそれぞれを指すポインタを格納し、各タスク記述ブロックは１つのパリティ演算についての情報を格納する。パリティ演算についてのデータは、異なるパリティ演算に使用される様々な複数の係数と同様に、メモリに格納される。各タスク記述ブロックは、下記を含め、パリティ演算ロジックが単一パリティ演算を実行するために使用する情報を格納する。すなわち、実行する演算のタイプ、タスク記述ブロックのサイズ、オプションであるタグ番号（一貫性チェックのため）、実行する演算のステータス結果のどれを格納するかについての応答待ち行列の指示、メモリに格納されているデータを指すポインタ、複数の係数のサブセットを指すポインタ、サブセット内の係数の数の指示、及びパリティ演算ロジックがパリティ演算の結果を格納するメモリ内の結果位置を指すポインタを格納する。この情報がすべてタスク記述ブロックに格納されると、タスク記述ブロックを指すポインタがタスク記述ブロック待ち行列に格納され、パリティ演算ロジックが要求するときはいつでもパリティ演算を実行することが可能になる。

【００２０】

【発明の実施の形態】＜パリティ演算＞図２を参照すると、本発明による冗長データ格納システム20は、データストライプ24を有する格納ディスク22を利用する。各データストライプ24は、複数のデータセグメントx_0～x_{N-1}と、少なくとも２つの対応するパリティセグメントＰ及びＱとを含む。Ｐ及びＱは、データセグメントx_0～x_{N-1}、第１のパリティ係数セットp_0～p_{N-1}、及び第２のパリティ係数セットq_0～q_{N-1}から導出される。パリティ係数は以下の式に従い、各データセグメントに対応する。

【００２１】

【数３】

$$P = p_0x_0 + p_1x_1 + p_2x_2 + p_{N-1}x_{N-1}$$

$$Q = q_0x_0 + q_1x_1 + q_2x_2 + q_{N-1}x_{N-1}$$

【００２２】本発明において、パリティ演算は、パリティセグメント生成演算、パリティセグメント再生成演算、及びデータセグメント再構築演算に分類することができる。

【００２３】パリティセグメント生成演算は、新しいデータストライプを作成する場合、すなわちパリティセグメントが全く新しいデータに基づいて作成される場合に行われる。

【００２４】パリティセグメント再生成演算は、新しいデータにより新しいデータセグメントが追加される場合、または読み出し／変更／書き込みサイクルにより１つまたは複数の既存のデータセグメントが変更される場合に、既存のストライプに対して行われる。パリティセ

グメント再生成演算では、パリティセグメントが、データストライプ全体を読み出すことなく増分的に変更される。例えば、新しいデータにより新しいデータセグメントx_4が追加されるものと想定する。P_{NEW}は、次のように計算される。

【００２５】

【数４】

$$P_{NEW} = P_{OLD} + p_4 x_4$$

【００２６】同様に、読み出し／変更／書き込みサイクルの結果、データセグメントx_2が変更されるものと想定する。この場合、P_{NEW}は、次のように計算される。

【００２７】

【数５】

$$P_{NEW} = P_{OLD} - p_2 x_{2OLD} + p_2 x_{2NEW}$$

【００２８】古いデータセグメント値から新しいP及びQの値を計算するために要するメモリは、ストライプが変更される都度初めからP及びQを計算するために要するメモリよりもはるかに少ない。

【００２９】本発明によれば、パリティセグメント再生成演算は、データセグメントの追加から生じるパリティ再生成、またはデータセグメントの変更から生じるパリティ再生成のいずれかにさらに分類することができる。

【００３０】データセグメント再構築演算は、２つの下位分類、すなわち、単一データセグメント再構築演算と、二重データセグメント再構築演算とを含む。生存データセグメントと組み合わせて、Pパリティセグメントまたは Qパリティセグメントのいずれかから、単一データセグメントを再構築することができる。一般に、データセグメントx_aは、次のようにしてパリティセグメントPまたはQのいずれかから再構築される。

【００３１】

【数６】

$$x_a = f(p_0, p_a)x_0 + f(p_1, p_a)x_1 + \ldots + f(p_a)P + \ldots + f(p_{N-1}, p_a)x_{N-1}$$

$$x_a = f(q_0, q_a)x_0 + f(q_1, q_a)x_1 + \ldots + f(q_a)Q + \ldots + f(q_{N-1}, q_a)x_{N-1}$$

式中、f( )は、使用しているパリティ生成コードに固有の適切な係数を生成する変換関数である。

【００３２】これらの式の実施形態の1つは、以下の通りである。

【００３３】

【数７】

$$x_a = p_a^{-1}(p_0x_0 + p_1x_1 + \ldots + P + \ldots + p_{N-1}x_{N-1})$$

$$x_a = p_a^{-1}(q_0x_0 + q_1x_1 + \ldots + Q + \ldots + q_{N-1}x_{N-1})$$

【００３４】生存データセグメントと組み合わせて、Pパリティセグメント及びQパリティセグメントから、２つのデータセグメントを再構築することができる。一般に、２つのデータセグメントx_a及びx_bは、次のようにしてパリティセグメントP及びQから再構築される。

【００３５】

【数８】

$$x_a = f(p_0, q_0, p_a, p_b, q_a, q_b)x_0 + f(p_1, q_1, p_a, p_b, q_a, q_b)x_1 + \ldots + f(p_a, p_b, q_a, q_b)P + f(p_k, q_k, p_a, p_b, q_a, q_b)x_k + f(p_{k+1}, q_{k+1}, p_a, p_b, q_a, q_b)x_{k+1} + \ldots + f(p_a, p_b, q_a, q_b)Q + \ldots + f(p_{N-1}, q_{N-1}, p_a, p_b, q_a, q_b)x_{N-1}$$

$$x_b = f(p_0, q_0, p_a, p_b, q_a, q_b)x_0 + f(p_1, q_1, p_a, p_b, q_a, q_b)x_1 + f(p_a, p_b, q_a, q_b)P + f(p_k, q_k, p_a, p_b, q_a, q_b)x_k + f(p_{k+1}, q_{k+1}, p_a, p_b, q_a, q_b)x_{k+1} + \ldots + f(p_a, p_b, q_a, q_b)Q + \ldots + f(p_{N-1}, q_{N-1}, p_a, p_b, q_a, q_b)x_{N-1}$$

式中、f( )は、使用しているパリティ生成コードに固有の適切な係数を生成する変換関数である。

【００３６】これらの式の実施形態の１つは、以下の通りである。

【００３７】

【数９】

$$x_a = (p, q_b \vdash p_b q_a)^{-1} ((q_b p_0 + p_b q_0)x_0 + (q_0 p_1 + p_0 q_1)x_1 + \ldots + q_b P + \ldots$$
$$+ (q_b p_k + p_b q_k)x_k + (q_b p_{k-1} + p_b q_{k+1})x_{k+1} + \ldots + p_b Q + \ldots \vdash$$
$$(q_b p_{N-1} + p_b q_{N-1})x_{N-1})$$
$$x_b = (p_a q_b + p_b q_a)^{-1} ((q_a p_0 + p_a q_0)x_0 + (q_a p_1 \vdash p_a q_1)x_1 + \ldots + q_a P + \ldots$$
$$+ (q_a p_k + p_a q_k)x_k + (q_a p_{k-1} + p_a q_{k+1})x_{k+1} + \ldots + p_a Q + \ldots \vdash$$
$$(q_a p_{N-1} + p_a q_{N-1})x_{N-1})$$

【００３８】一般に、上記パリティ演算はすべて、有限個のかかる係数を有するベースセットから選択される既知の係数の種々の組み合わせを用いることで達成することができる。これら係数は、p_0～p_{N-1}、q_0～q_{N-1}、及び変換関数ｆ(　)から得られる係数を含む。任意の特定のパリティ演算は、計算中の実際のデータセグメントまたはパリティセグメントに応じて、これら係数のサブセットを利用する。特定の計算に必要な特定の係数サブセットは、演算の分類及び関係する特定のデータセグメントやパリティセグメントの双方に依存する。従って、所与のパリティ演算の分類内には種々の状況すなわちシナリオがあり、それぞれに異なる係数サブセットが必要となる。例えば、データセグメントx_5をストライプに追加する場合、係数p_5及びq_5が必要とされる場合に１つのシナリオが発生する。データセグメントx_6をストライプに追加する場合、係数p_6及びq_6が必要な場合に別のシナリオが発生する。

【００３９】＜係数サブセット＞図３は、複数の係数サブセット31を含むメモリアレイ30を示している。各係数サブセットは、対応するデータセグメントに適用されパリティ計算結果を生成する、予め選択された、かつ／または予め計算された係数のリストまたは連結である。本発明によれば、異なる係数サブセットが予め選択され、各異なる演算シナリオについて格納される。次に、パリティ演算ロジックによる参照及び直接の使用のため、サブセットはフォーマットされ線形メモリアレイにパックされる。異なるシナリオには異なる数の係数が必要であるため、サブセットは同じ長さまたはサイズである必要はない。

【００４０】本発明の実施形態において、各係数は単一バイトである。「パックされる」という語は、好ましくは使用していない空間に干渉することなく、係数のサブセットまたはストリングが線形メモリ内で連結され、記憶空間を節約することを意味する。

【００４１】ここで、パリティ演算を行う場合、サブセット内の係数とデータのセグメント（データセグメントまたはパリティセグメントのいずれか）との間には１対１の対応がある。各係数は、対応するデータセグメントまたはパリティセグメントにのみ適用されて、演算結果を生成する。

【００４２】可能な各パリティ計算のケースすなわちシナリオにつき、１つの係数サブセットがアレイに含まれる。特定の各計算シナリオに関するアレイ内のサブセットの開始位置を見つけるために、固有のインデックス式を用いることができる。一般に、サブセットは、Ｐ及びＱそれぞれに関わる計算に対応する対で構成される。

【００４３】図３を参照すると、メモリアレイ30は、複数の分類グループ32、33、34、35、及び36を含み、各分類グループは、特定のパリティ演算分類に対応する係数サブセット31を含む。分類グループにおける各サブセットは、グループの分類内で発生する特定のシナリオに関する係数を有する。１つの例外を除き、係数サブセットは任意の所与の分類グループ内で同一のサイズである。

【００４４】アレイの開始位置からグループの開始位置までのグループオフセットを計算することで、アレイ30内で特定の分類グループを見つけることができる。このグループ・オフセットは、グループのアレイへのベースインデックスである。分類グループ内で特定の係数サブセットを見つけるために、グループの開始位置からのサブセット・オフセットがベースインデックスに加えられる。これにより、所望の係数サブセットの開始位置を見つけるアレイ内へのインデックスが生成される。

【００４５】本発明の一実施形態によれば、一般的なパリティ演算分類は以下のように定義される。

【００４６】１）パリティ生成演算―既存のデータまたはパリティを持たない部分ストライプまたは完全に新しいストライプ。

【００４７】２）セグメントの追加によって生じるパリティ再生成演算―新しいデータセグメントを２つのパリティセグメントに組み込むことによる、ストライプの増分的成長。

【００４８】３）セグメントの変更によって生じるパリティ再生成演算―すでに２つのパリティセグメントに組み込まれているデータセグメントの変更（読み出し／変更／書き込み）。

【００４９】４）単一データセグメント再構築―２つのパリティセグメントの一方とストライプからの生存データセグメントとを使用した単一データセグメントの再構築。２つの記憶装置に障害が発生する場合に、障害が発生した２つの記憶装置の一方がＰまたはＱを保持するこ

とができるため、Ｐパリティセグメントまたは Ｑパリティセグメントの何れかからの再構築がサポートされる。

【００５０】５）二重データセグメント再構築―２つのパリティセグメントＰ及びＱと、ストライプからの生存データセグメントとを用いたストライプの２つのデータセグメントの再構築。

【００５１】＜分類１の係数サブセットの構造＞アレイの第１の分類グループ32は、パリティ生成演算のための係数サブセットを含む。パリティ生成演算は、新しいＰセグメント及びＱセグメントを新しいデータセグメント x_0 ～ x_{N-1} から生成する。この分類グループには、２つの係数サブセットしかない。サブセットはそれぞれ、パリティセグメントＰ及びＱの生成に対応している。

【００５２】

【数１０】

$$P: \quad \{p_0, p_1, \ldots p_{N-1}\} \text{ and}$$

$$Q: \quad \{q_0, q_1, \ldots q_{N-1}\}$$

これらサブセットはそれぞれ同じ長さ（Ｎ）である。

【００５３】＜分類２の係数サブセットの構造＞アレイの第２の分類グループ33は、ストライプを増分的に追加するパリティ演算のための係数サブセットを含む。このタイプの演算は、任意の所与の新しいまたは追加のデータセグメント x_a ～ x_b （但し、ｂ＜Ｎ、かつａ≦ｂ）の連続範囲と組み合わせてＰ及びＱの各セグメントを更新する。データセグメント０～Ｎ－１内のデータセグメントのあらゆる可能な範囲ａ～ｂに対応する、これら演算の異なるシナリオが多数ある。各シナリオには、異なる係数サブセットが必要である。例えば、新しいまたは追加のデータセグメントが x_3 及び x_4 である場合、Ｐの計算に必要な係数サブセットは｛P_3，P_4｝である。新しいまたは追加のデータセグメントが x_2 ～ x_5 である場合、Ｐの計算に必要な係数サブセットは｛P_2，P_3，P_4，P_5｝である。データセグメント０～Ｎ－１内の可能な範囲の総計は、Ｎの値に依存する。

【００５４】分類グループ２の各係数サブセットは、サブセットがＰの計算に適用されるか、またはＱの計算に適用されるかを示す２つの初期パラメータを含む。これら初期パラメータはそれぞれ「０」または「１」のいずれかに設定される。これら係数のうちの最初の値「１」は、計算がパリティセグメントＰに関わることを示す。これら係数の二番目の値「１」は、計算がパリティセグメントＱに関わることを示す。これら２つのパラメータの一方のみが随時「１」に等しく設定されるべきである。

【００５５】分類２サブセット内の残りの係数は、新しく追加されたデータストライプからＰ及びＱを再生成するために使用される係数の部分範囲（sub-range）である。従って、この分類グループは以下の形態の複数の係数サブセットを含む。

【００５６】

【数１１】

$$P: \quad \{1, 0, p_a, \ldots p_b\} \text{ and}$$

$$Q: \quad \{0, 1, q_a, \ldots q_b\}$$

【００５７】分類グループ33は、Ｎに応じて、より広い範囲の０～Ｎ－１内のあらゆる範囲ａ～ｂに対応するような複数のサブセットを含む。アレイのこのセクション内の係数サブセットの長さすなわちサイズは可変であり、各演算シナリオについて（ｂ－ａ）に等しい。

【００５８】この分類グループ内において、係数サブセットは、長さによって配列されグループ化される。すなわち、最小数の係数を含む係数サブセットは、分類グループの最初の部分に配置される。最大数の係数を含む係数サブセットは、分類グループの最後に配置される。これらの各グループの中で、係数サブセットは、係数サブセットでカバーされる範囲の係数の下付文字に従った順に配列される。従って、ａ＝０を有するサブセットが最初に配置され、ａ＝１を有するサブセットが次に配置され、以下同様である。

【００５９】＜分類３の係数サブセットの構造＞第３の分類グループ34内の係数サブセットは、単一データセグメントが変更されるとき、Ｐ及びＱを更新するために使用される。このタイプの演算は、データセグメント x_a が変更される場合に、Ｐセグメント及びＱセグメントを更新する。

【００６０】分類２のグループのように、各分類３サブセットの最初の２つのパラメータは、グループの係数がＰの計算またはＱの計算に適用可能であるか否かを示す。これら係数はそれぞれ「０」または「１」に設定される。これら係数の最初の値が「１」であることは、サブセットの係数がパリティセグメントＰに適用されることを示す。これら係数の二番目の値が「１」であることは、サブセットの係数がパリティセグメントＱに適用されることを示す。

【００６１】各サブセットは、変更中のデータセグメント x_a に対応する単一の残りの係数を含む。

【００６２】

【数１２】

$$P: \quad \{1, 0, p_a\} \text{ and}$$

$$Q: \quad \{0, 1, q_a\}$$

【００６３】第３の分類グループ34は、０からＮ－１までのａのすべての値に対応するＮ対のこのようなサブセットを含む。これらサブセットは、ａ＝ｂである分類２係数サブセットの特別な場合に対応するため、新しい単一データセグメントをストライプに追加する際に使用可能なことに留意されたい。

【００６４】＜分類４の係数サブセットの構造＞第４の分類グループ35内の係数サブセットは、パリティセグメントの一方及び生存データセグメントに基づいて単一データセグメントx_aを再構築するために使用される。係数は、選択された誤り修正コードの数学的処理（ｆ（　））に従って変換されて再構築演算を実行することを除き、分類１の係数に密接に対応する。

【００６５】

【数１３】

$$P: \quad \{ f(p_0, p_a), f(p_1, p_a), \ldots, f(p_a), \ldots, f(p_{N-1}, p_a) \}$$

$$Q: \quad \{ f(q_0, q_a), f(q_1, q_a), \ldots, f(q_a), \ldots, f(q_{N-1}, q_a) \}$$

より具体的には、以下の通りである。

【数１４】

【００６６】

$$P: \quad (p, q_b + p_b q_a)^{-1} ((q_b p_0 + p_b q_0), (q_0 p_1 + p_0 q_1), \ldots, q_b P, \ldots, (q_b p_k + p_b q_k), (q_b p_{k-1} + p_b q_{k+1}), \ldots, p_b Q + \ldots + (q_b p_{N-1} + p_b q_{N-1}))$$

$$P: \quad (p_a q_b + p_b q_a)^{-1} ((q_a p_0 + p_a q_0), (q_a p_1 + p_a q_1), \ldots, q_a P + \ldots + (q_a p_k + p_a q_k), (q_a p_{k-1} + p_a q_{k+1}), \ldots, p_a Q, \ldots, (q_a p_{N-1} + p_a q_{N-1}))$$

【００６７】第４の分類グループは、０からＮ－１までのａのすべての値に対応するＮ対のこのようなサブセットを含む。各サブセットにおいて、係数ｆ（p_a）または係数ｆ（q_a）はデータセグメントx_aに対応することに留意されたい。

【００６８】＜分類５の係数サブセットの構造＞第５の分類グループ36内の係数サブセットは、２つのパリティセグメント及び生存データセグメントに基づいて、２つのデータセグメントx_a及びx_bを再構築するために使用される。係数は、選択された誤り修正コードの数学的処理（ｆ（　））に従って変換されて再構築演算を実行することを除き、分類１の係数に密接に対応する。

【００６９】

【数１５】

$$x_a: \quad \{ f(p_0, q_0, p_a, p_b, q_a, q_b), f(p_1, q_1, p_a, p_b, q_a, q_b), \ldots, f(p_a, p_b, q_a, q_b), \ldots, f(p_k, q_k, p_a, p_b, q_a, q_b), f(p_{k+1}, q_{k+1}, p_a, p_b, q_a, q_b), \ldots, f(p_a, p_0, q_a, q_b), \ldots, f(p_{N-1}, q_{N-1}, p_a, p_b, q_a, q_b) \}$$

$$x_b: \quad \{ f(p_0, q_0, p_a, p_b, q_a, q_b), f(p_1, q_1, p_a, p_b, q_a, q_b), \ldots, f(p_a, p_b, q_a, q_b), \ldots, f(p_k, q_k, p_a, p_b, q_a, q_b), f(p_{k+1}, q_{k+1}, p_a, p_b, q_a, q_b), \ldots, f(p_a, p_0, q_a, q_b), \ldots, f(p_{N-1}, q_{N-1}, p_a, p_b, q_a, q_b) \}$$

【００７０】アレイの５番目のセクションは、０からＮ－１の範囲内のａ及びｂのあらゆる可能な組み合わせに対応する、（Ｎ×（Ｎ－１））／２対のかかるサブセットを含む。各サブセットにおいて、係数ｆ（p_a，p_b，q_a，q_b）は、データセグメントのうちどれが再構築されているかに応じて、データセグメントx_aまたはx_bに対応することに留意されたい。

【００７１】これらの式の可能な実施形態の１つは、以下の通りである。

【００７２】

【数１６】

$$x_a: \quad (p_a, q_b + p_b, q_a)^{-1}(q_b, p_0 + p_b, q_0), (p_a, q_b + p_b, q_a)^{-1}(q_b, p_1 + p_b, q_1), \ldots, (p_a, q_b + p_b, q_a)^{-1} q_b, \ldots, (p_a, q_b + p_b, q_a)^{-1}(q_0, p_k + p_0, q_k), (p_a, q_b + p_b, q_a)^{-1}(q_0, p_{k+1} + p_0, q_{k+1}), \ldots, (p_a, q_b + p_b, q_a)^{-1} p_b, \ldots, (p_a, q_b + p_b, q_a)^{-1}(q_0, p_{N-1} + p_0, q_{N-1})$$

$$x_b: \quad (p_a, q_b + p_b, q_a)^{-1}(q_b, p_0 + p_b, q_0), (p_a, q_b + p_b, q_a)^{-1}(q_a p_1 + p_a, q_1), \ldots, (p_a, q_b + p_b, q_a)^{-1} q_a, \ldots, (p_a, q_b + p_b, q_a)^{-1}(q_a, p_k + p_a, q_k), (p_a, q_b + p_b, q_a)^{-1}(q_a, p_{k+1} + p_a, q_{k+1}), \ldots, (p_a, q_b + p_b, q_a)^{-1} p_a, \ldots, (p_a, q_b + p_b, q_a)^{-1}(q_a, p_{N-1} + p_a, q_{N-1})$$

【００７３】＜係数サブセットの使用＞図４は、上記アレイ格納方式に従ってパリティ演算を実行する方法を示す。第１のステップ100で、種々のパリティ演算をパリティセグメント生成演算、パリティセグメント再生成演算、及びデータセグメント再構築演算を含む分類に分類する。より具体的には、演算は、パリティ生成演算、セグメント追加によって生じるパリティ再生成演算、セグメントの変更によって生じるパリティ再生成演算、単一データセグメント再構築演算、または二重データセグメント再構築演算のいずれかに分類される。パリティ演算の各分類は、複数の多様な分類シナリオを含み、その各シナリオは各パリティ係数サブセットに関係する。

【００７４】ステップ102で、個々のパリティ係数を予め計算し、異なるパリティ演算及びパリティ演算の異なるシナリオに使用するパリティ係数サブセットを予め選択する。このステップは、すでに述べた説明に従って行われる。

【００７５】ステップ104で、予め選択したパリティ係数サブセットのすべてを、インデックスの付いた線形メモリアレイに格納する。サブセットには、パリティ計算ロジックによりアクセス可能である。このステップは、係数サブセットを予めフォーマットして、ハードウェアベースのパリティ演算ロジックが効率的に利用できるようにすることを含む。特に、各サブセットの個々の係数は隣接するバイトまたは記憶装置にパックされ、ハードウェアベースのパリティ演算ロジックに固有の様式に配列される。このステップの結果、メモリアレイは、種々の計算シナリオそれぞれに対応する単一の係数サブセットを含むようになる。

【００７６】個々の係数及び係数サブセットは、非干渉データ要素を用いてパックされる。アレイのサブセットはグループ化され、すでに述べたように順序づけられ、係数サブセットはそれらの分類順によって分類グループにグループ化される。第２の分類グループ内では、サブセットは可変サイズを有する。さらに、第２の分類グループ内のサブセットは、サイズによりサブグループ化され、最小の番号を有する係数に従い昇順に順序づけられる。

【００７７】パリティ演算中、パリティ演算ロジックがメモリアレイにアクセスし、パリティ演算の種々のシナリオでの使用に適した係数サブセットを得る。よって、ステップ106で、格納されているパリティ係数サブセットのうちどれが特定のパリティ演算に必要なのかを決定する。このステップは、パリティ演算の分類と線形メモリアレイへのグループオフセットの決定を含み、そのパリティ演算分類に対応する分類グループの開始位置を示す。次に、グループ内への所望の係数サブセットの位置までのサブセットオフセットが計算される。

【００７８】ステップ106は、第２の分類グループ以外は単純である。詳細に上述したように、第２の分類グループは長さすなわちサイズが可変の係数サブセットを含み、特定の係数サブセットのオフセットを決定することが困難である。しかし、本願発明者らは、第２の分類グループが上述したように配列されるとき、同じサイズの係数サブセットのサブグループを順序づけると、特定のサブグループに対するオフセットを、係数サブセットのサブグループのサイズ及びＮ（任意のサブグループに含まれる最大数の係数）の関数として計算することができることを見出した。具体的には、サブセットサイズL_iに対応する特定のサブグループｉに対するオフセットは、以下に等しい。

【００７９】

【数１７】

$$((L_i - 1)(12N + L_i(3N - 2L_i - 5))/6) - 3(N - 1)$$

【００８０】この式は、各サブセットには、Ｐ及びＱに対応する定数の対（上述）が存在するものと想定している。しかし、Ｌは（ｂ－ａ）に等しい。サブグループｉ内において、特定の係数サブセットのオフセットはａ（L_i＋２）に等しい。従って、x_a～x_bに対応する係数の範囲に関して、分類グループへの全体的なオフセットは、次のようになる。

【００８１】

【数１８】

$$(((L_i - 1)(12N + L_i(3N - 2L_i - 5))/6) - 3(N - 1)) + a(L_i + 2)$$

第２の分類グループのサイズは、次の式によって与えられる。

【００８２】

【数１９】

$$((N - 1)(12N + N(3N - 2N - 5))/6) - 3(N - 1)$$

【００８３】メモリアレイへの適切なオフセットを決定した後、決定したパリティ係数サブセットをメモリから読み出すステップ108が行われる。ステップ110で、メモリから読み出されたパリティ係数サブセットを用いて、特定のパリティ演算を実行する。

【００８４】＜ディスクコントローラの動作＞図５は、本発明によるディスクコントローラ200の最も適切な構成要素を示す。ディスクコントローラ200は、マイクロプロセッサ201及び関連するメモリ202を有する。さらに、ディスクコントローラ200は、ハードディスクインタフェースコンポーネント203及び通信コンポーネント204を備える。ハードディスクインタフェースコンポーネント203は、ディスクコントローラ200に接続されて制御されるハードディスクへのアクセス手段を提供する。通信コンポーネント204は、ホストコンピュータとハードディスクコントローラ200の間のインタフェースとして機能する。

【００８５】これらコンポーネントに加え、ハードディスクコントローラ200は、特定用途向け集積回路（ＡＳ

ＩＣ）の形態のハードウェアベースのパリティ演算ロジック205を備える。「ハードウェアベース」という語は、このロジックコンポーネントが、ソフトウェアベースのロジックとは対照的に、プログラムメモリからの命令の検索や実行を行わないことを意味する。むしろ、ロジックコンポーネントは、信号及びデータを処理する専用の相互接続されるロジック要素を有する。このようなハードウェアベースのロジックはマイクロプロセッサまたは他の命令ベースのプロセッサよりも柔軟性が低いが、ハードウェアベースのロジックは命令ベースのロジックよりもはるかに高速であることが多い。

【００８６】一般に、ディスクコントローラは以下のように動作する。マイクロプロセッサ201は、ホストコンピュータとの通信を処理し、ホストコントローラへの、かつホストコントローラからのすべてのデータ転送を調整する。さらに、マイクロプロセッサ201は、実際のディスク転送をすべて調整する。しかし、データは、ディスクに書き込まれる前にメモリ202にバッファリングされる。パリティ演算は、マイクロプロセッサ201の制御下でメモリ202内のデータに対して行われる。

【００８７】初期化中、マイクロプロセッサ201は、メモリ202内に係数サブセットテーブル212を構築する。続いて、パリティ演算のときになると、マイクロプロセッサ201は、パリティ演算の分類及びシナリオを決定する。この情報がいったん決定されると、マイクロプロセッサ201は、パリティ演算の目的となる１つまたは複数のデータセグメント及びパリティセグメント（まとめてデータブロック214と呼ぶ）の、メモリ202における位置を示すスクリプトを作成する。スクリプトは、パリティ演算に適切な係数サブセットが見つかる係数サブセットテーブル212へのオフセット、及び係数サブセットに含まれる係数の数を示す。スクリプトはまた、要求された計算の結果が配置されるメモリ202内の位置も示す。各スクリプトは、単一パリティ演算についての情報を格納し、かかるスクリプトを格納するためのメモリ構造は、本明細書においてタスク記述ブロック（ＴＤＢ）と呼ばれる。ＴＤＢはメモリ202内の特定の位置に格納され、その位置（例えば、３２ビットアドレス）へのポインタはメモリ202内のＴＤＢ待ち行列216に格納される。応答待ち行列218もまたメモリ202に格納され、ハードウェアロジック205がこれを用いて、パリティ演算が成功したか否かについての指示をマイクロプロセッサ201に戻す。

【００８８】スクリプトがメモリに配置されると、待ち行列216におけるスクリプトに対するＴＤＢへのポインタが存在することで、ハードウェアロジックに通知される。ハードウェアロジックはこれに応答して、（ａ）指示された係数、データセグメント、及びパリティセグメントを検索し、（ｂ）指示された係数に基づいて適切なパリティ演算を行い、（ｃ）データ及び／または計算したパリティセグメントをメモリに戻し、（ｄ）エントリを応答待ち行列２１８に書き込むことで、演算が首尾良く完了したか否かをマイクロプロセッサ２０１に知らせる。

【００８９】ハードウェアロジックは、係数及びデータ／パリティセグメントの積を合計することで、各種の異なるパリティ演算を行うように構成される。異なる演算は実際に、係数、データセグメント、及びパリティセグメントの数及び選択においてのみ異なる。これら変数は、スクリプトによって特定される。このように、この演算は、ハードウェアベースの計算に非常に都合が良い。

【００９０】図６は、タスク記述ブロック250の例示的な構造をさらに詳細に示す。本明細書において、ＴＤＢ250は、ＴＤＢ250内の情報に基づいてパリティ演算を行うために、ハードウェアロジック205によりアクセスされることに関連して説明される。代わりに、図６に示すメモリ構造は、種々の方法で（例えば、ソフトウェアまたはファームウェアを実行するマイクロプロセッサまたは他の命令ベースのプロセッサにより）パリティ演算を実行する他のシステムで使用することもできる。

【００９１】ＴＤＢ待ち行列216は複数のエントリを含み、各待ち行列エントリは、要求ヘッダ及びＴＤＢポインタの両方を含む。ＴＤＢポインタ252はＴＤＢ250をポイントするが（例えば、ＴＤＢ250の開始アドレスをメモリ202に格納する）、対応する要求ヘッダ251は、ＴＤＢ250に基づいて実行される要求された動作についての様々な情報を格納する。一実施形態においては、ＴＤＢポインタ252及び要求ヘッダ251の両方が３２ビット値であり、要求ヘッダ２５１は以下の情報を含む。すなわち、ＴＤＢ20のサイズ（例えば、８バイトダブルワードグループで）、ハードウェアロジック205がＴＤＢ250内のデータに基づいて実行すべき演算の指示、タグベースの一貫性チェック（より詳細に後述する）を実行すべきか否か、演算完了情報を書き込むべき応答待ち行列218の識別、及びタグ番号（タグベースの一貫性チェックが実行中である場合）である。

【００９２】ＴＤＢ250は、ヘッダ253と、複数のディスクデータポインタ254と、出力ポインタ256と、係数ポインタ及びデータ258とを含む複数のフィールドを有する。ヘッダ253は、ＴＤＢについての様々な情報を含み、一実施形態では以下を含む。すなわち、ＴＤＢ250のサイズ（例えば、８バイトダブルワードグループで）、ハードウェアロジック205がＴＤＢ250内のデータに基づいて実行すべき演算の指示（例えば、上記パリティ演算の５つの分類のうちの１つの識別）、ハードウェアロジック205がＴＤＢ250内のデータに基づいて実行すべき演算の実行優先度（例えば、所望であれば、異なるタイプの演算に異なる実行優先度を割り当てることができる）、及びタグ番号（タグベースの一貫性チェックが

実行中である場合）である。

【００９３】各ディスクデータポインタ254は、パリティ演算に関するディスクデータのメモリ202内の位置をポイントする（図５のデータブロック２１４のうちの１つをそれぞれポイントする）。実行中のパリティ演算に応じて、各ポインタ254によってポイントされるデータは、パリティ演算を行うべきストライプのデータセグメントまたはパリティセグメントである。ポインタ254は、ハードウェアロジック205により（ストライプにおいて最も小さい番号の付いたデータセグメントから最も大きな番号の付いたデータを配列し、その後にＰパリティセグメント及びＱパリティセグメントが続くといった）既知の順に従って配列されるため、ハードウェアロジック205は、パリティ演算についてどのメモリ位置にどのセグメントデータがあるかを知る。出力ポインタ256は、パリティ演算ロジック205がパリティ演算の結果を書き込むべきメモリ202内の位置260を指すポインタである。実行中のパリティ演算のタイプに応じて、演算の結果はＰパリティ値及びＱパリティ値のいずれか（例えば、パリティ生成演算またはパリティ再生成演算の場合）、またはデータセグメント値（例えば、データセグメント再構築の場合）であることができる。単一の出力ポインタ256を図６に示しているが、代わりに複数の出力ポインタをＴＤＢ250に含めてもよい（例えば、Ｐ値に１つのポインタ、Ｑ値に１つのポインタというように、複数の値が生成される場合）。

【００９４】係数ポインタ及びデータ258は、パリティ演算ロジック205が使用すべき係数サブセットが位置する、係数テーブル30内の位置を指すポインタを含む。ポインタ及びデータ258内には、係数テーブルから検索すべき係数の数の指示も存在し得る（これは、ヘッダ253において識別される実行すべきパリティ演算のタイプの指示とは別のものであってもよく、また代わりに実行すべきパリティ演算のタイプの指示に埋め込んでも（または固有であっても）よい）。従って、係数ポインタ及びデータ258は（ヘッダ253と組み合わせて）、どこでパリティ演算に関する係数を得るか、及びどのくらい係数を得るべきか、ならびにどのパリティ演算を実行すべきかという必要な情報をパリティ演算ロジック205に与える。

【００９５】必要な情報がすべてパリティ演算のＴＤＢ250に格納されると、マイクロプロセッサ201は、そのＴＤＢを指すポインタ及び対応する要求ヘッダ情報を含むエントリをＴＤＢ待ち行列216に追加する。待ち行列216に演算を加え、必要なデータが適切な位置で利用できるようになるまで待ち行列216へのエントリの追加を行わないことにより、パリティ演算ロジック205は、確実にパリティ演算の実行に必要な情報を有する。従って、パリティ演算ロジック205は、他のパリティ演算が待ち行列で待機している間、パリティに関する追加情報を待つことで遅延されず、ロジック205の高速演算が可能になる。

【００９６】パリティ演算を実行する前に、パリティ演算ロジック205は、一貫性チェックをオプションとして実行して、正しいＴＤＢを使用していることを検証することができる。一実施形態では、この一貫性チェックは、演算のタイプ及びＴＤＢのサイズの検証に基づく。この実施では、ＴＤＢ250のヘッダ253は、上記と同様に、実行すべき演算のタイプの指示及びＴＤＢ250のサイズを含む。正しいＴＤＢにアクセスしていることを検証するために、ハードウェアロジック205は、要求ヘッダ251内の演算のタイプ及びＴＤＢサイズにアクセスし、これらがヘッダ253に格納されている演算のタイプ及びＴＤＢサイズに一致するか否かをチェックする。ハードウェアロジック205が、２つの演算タイプ及び２つのＴＤＢサイズが一致すると判断した場合、ＴＤＢ250は正しいＴＤＢであると検証され、その他の場合、ＴＤＢは正しいＴＤＢではなく、ハードウェアロジック205は、（例えば、応答待ち行列218内のエントリを介して）コマンド不一致のためにパリティ演算が失敗したことをマイクロプロセッサ２０１に知らせる。

【００９７】一貫性チェックはさらに行うことが可能であり、本明細書においてこれはタグベースの一貫性チェックと呼ばれる。タグベースの一貫性チェックを用いる場合、マイクロプロセッサ201はタグ番号を演算に割り当て、タグ番号は要求ヘッダ251及びヘッダ253の両方に格納される。ハードウェアロジック205は、演算タイプ及びＴＤＢサイズを検証することに加え、ヘッダ251及び253におけるタグ番号の一致もチェックする。タグ番号が一致する場合、ＴＤＢ250は正しいＴＤＢであると検証され（演算タイプ及びＴＤＢサイズも一致するものと仮定する）、その他の場合は、ＴＤＢは正しいＴＤＢではなく、ハードウェアロジック205は、（例えば、応答待ち行列218内のエントリを介して）タグ不一致によりパリティ演算が失敗したことをマイクロプロセッサ201に知らせる。

【００９８】ハードウェアロジック205が演算を完了すると、計算した演算結果が、出力ポインタ256によってポイントされるメモリ位置260に格納される。さらに、ハードウェアロジック205によって実行された演算のステータスを示すエントリが応答待ち行列218に追加される。複数の応答待ち行列218が存在する場合があり、ハードウェアロジック205がエントリを追加する適切な応答待ち行列は、演算に対応する要求ヘッダ251で示される。

【００９９】各応答待ち行列218は複数のエントリを含むことができ、各エントリは応答ヘッダ及びＴＤＢポインタを含む。ＴＤＢポインタ262は、ＴＤＢ250をポイントするが（例えば、ＴＤＢ250の開始アドレスをメモリ202に格納する）、対応する応答ヘッダ261は、ＴＤＢ250

に基づいて実行された演算についての様々な情報を格納する。一実施形態では、ＴＤＢポインタ262及び応答ヘッダ261の両方が３２ビット値であり、応答ヘッダ261は以下の情報を含む。すなわち、ＴＤＢ250のサイズ（例えば、８バイトダブルワードグループで）、ハードウェアロジック205がＴＤＢ250内のデータに基づいて実行した演算の指示、タグベースの一貫性チェックが実行されたかどうか、二重パリティブロック計算の結果、ゼロしか含まないＰパリティブロック及びＱパリティブロックが生じたり、または単一パリティブロック計算の結果ゼロしか含まないＰパリティブロックが生じる場合に設定される「パリティ・オール・ゼロ」フラグ（このフラグは、演算がパリティ演算ではない場合でも設定される）、ソースデータが定義されたフォーマットパターンに一致する場合に設定される「パターン一致」フラグ、演算を実行する際にどのタイプのエラーが（もしあれば）発生したかに関する指示、及びタグ番号（タグベースの一貫性チェックが実行中である場合）を含む。一実施形態では、応答ヘッダ261を介して以下のエラーを示すことができる。すなわち、タグ不一致（待ち行列216のヘッダ251内のタグが、ＴＤＢ250のヘッダ253内のタグと一致しない）、コマンド不一致（ヘッダ251における演算タイプ及びＴＤＢサイズが、ヘッダ253内の演算タイプ及びＴＤＢサイズに一致しない）、パリティエラー（メモリ202から戻されるデータにおいて検出されたパリティエラー）、メモリテストデータ不一致（miscompare）（パリティ演算ではなくメモリテスト演算が実行中である場合、これは、メモリテスト中メモリ202から戻されるデータが不正確であったことを示す）、及び不正コマンド（ヘッダ251及びヘッダ253で示される演算タイプが正当な値ではない）である。

【０１００】従って、図６に示すメモリ構造を使用して、単一ＴＤＢを用いてパリティ演算ロジック205がパリティ演算を実行するために必要な情報をすべて識別する。単一ポインタを用いて各ＴＤＢを識別することができ、複数のパリティ演算を図５のＴＤＢ待ち行列216に容易に入れることが可能になる。次いで、パリティ演算ロジック205は、演算を待ち行列216から取り除くことにより、パリティ演算を特定の様式で（例えば、先入れ先出し（ＦＩＦＯ）方式で）処理することができる。

【０１０１】さらに、図５及び図６に示すように、どの演算を実行する必要があるかについての記述は、作業を実際に実行するコンポーネントから分離される。この分離により、２つの異なるプロセス（何をすべきかを記述するプロセス及びする必要があることを実際に実行するプロセス）を、他方がどのように実行されているか（あるいは、どのコンポーネントまたはエンティティがそのプロセスを実行しているか）に関わらず、互いに独立して動作することが可能になる。従って、図５のマイクロプロセッサ201は、どのコンポーネントが実際にパリティ演算を実行しているのかに関わらず（例えば、パリティ演算は、ハードウェアロジック205または代替として他のコンポーネント（ソフトウェア命令を実行する別のマイクロプロセッサ等）によって実行することができる）、パリティ演算についての必要な情報を準備することができる。同様に、ハードウェアロジック205は、パリティ演算についての情報をメモリ202から獲得でき、どのエンティティが情報をメモリ202内に準備するか、または情報がどのように得られたかに関わらず、パリティ演算を実行することができる。

【０１０２】図７は、上記メモリ構造によるパリティ演算についてのタスク記述ブロックを使用する方法を示す。図７の方法は、図５のマイクロプロセッサ201及びハードウェアロジック205によって実行される動作を示す（マイクロプロセッサ201によって実行される動作は左側にあり、ハードウェアロジック205によって実行される動作は右側にある）。第１のステップ270で、実行するパリティ演算のＴＤＢを生成する。この生成は、様々な従来の方法の任意のもので実行することができる（例えば、ＴＤＢにメモリを割り当てる、既に割り当てられたが使用されていないメモリを使用するなど）。ステップ272で、コマンドタイプ、応答待ち行列識別子、及びタグ番号（オプション）をＴＤＢのヘッダに格納する。他の情報（上述）がヘッダに含まれてもよい。ステップ274で、パリティ演算に関するディスクデータメモリ位置を指すポインタをＴＤＢに格納し、ステップ276で、パリティ演算に関する結果メモリ位置を指すポインタをＴＤＢに格納する。ステップ278で、使用に適した係数サブセットを決定し（上記図４のステップ106と同様）、ステップ280で、その係数サブセットを指すポインタをＴＤＢに格納する。ステップ282で、パリティ演算ロジックにどのパリティ演算を実行すべきかを知らせるために、パリティ演算分類の指示をＴＤＢに格納する。これらの情報すべてがＴＤＢに格納されると、ステップ284で、ＴＤＢ待ち行列に、要求ヘッダ及びこの新たに追加されたＴＤＢを指すポインタを含むエントリを追加する。従って、ＴＤＢの指示は、パリティ演算についてのすべてのデータがメモリ202内で利用できるようになるまで、ＴＤＢ待ち行列に追加されず、これによって、パリティ演算ロジックがメモリ202にアクセスし、所望のときにいつでもパリティ演算を実行することが可能になる。

【０１０３】ＴＤＢ待ち行列エントリが追加された後のある時点において、ステップ286で、ハードウェアロジック205が要求された演算を実行する。演算が完了または失敗すると、ステップ288で、演算が首尾良く完了したか、それともエラーが発生したかを示す（上記ステップ272において識別されるような）応答エントリを適切な応答待ち行列に追加する。

【０１０４】図７の説明は、ＴＤＢエントリ（要求ヘッ

ダ及びＴＤＢポインタの両方）が同時にＴＤＢ待ち行列に追加される（ステップ284）ことを述べている。代わりに、情報が利用可能な場合はＴＤＢ待ち行列内の要求ヘッダを埋めても良く、この場合は、ステップ284で、ＴＤＢポインタだけを待ち行列エントリに追加する（及び本質的にＴＤＢ待ち行列エントリが完全に埋められたことをハードウェアロジック205に通知する）。

【０１０５】本発明について、主に異なるタイプのパリティ演算を冗長ディスクドライブシステムにおいて実行することに関連して説明したが、本発明は、代替的に他の状況で使用することができる。例えば、本発明は、ハードウェア加速メモリテストを実行するために使用することも可能であり、この場合、検証のために書き込まれるべきメモリアドレスは、図６のフィールド254によって識別され、図５のハードウェアロジック205がテストデータを識別されたメモリアドレスに書き込み、そのメモリアドレスからデータを読み戻し、読み戻されたデータをポインタ256によって識別される位置に格納する。

【０１０６】別の例として、本発明は、コピー機能を実行するために使用することができる。この場合、コピーするデータはポインタ254によって識別され、得られるコピーの位置はポインタ256によって識別され、ハードウェアロジック205がコピーを実行する。

【０１０７】さらに別の例において、本発明は、パターンマッチングを実行するために使用することもできる。この場合、比較されるデータセグメントはポインタ254によって識別され、ハードウェアロジック205が比較を実行し、比較の結果（例えば、２つのデータセグメントが一致するかどうかの指示）をポインタ256によって識別される位置に書き込む。

【０１０８】さらに別の例として、本発明は、チェックサム計算の実行に使用することもできる。この場合、チェックサムが計算されるデータセグメントがポインタ254によって識別され、ハードウェアロジック205がチェックサムを計算し、得られるチェックサム値をポインタ256によって識別される位置に書き込む。

【０１０９】本発明には例として以下の実施形態が含まれる。

【０１１０】１．冗長データ記憶システムにおいてパリティ演算を実行するディスクコントローラ（200）であって、実行すべき前記パリティ演算についての情報をそれぞれ識別する複数のタスク記述ブロック（250）を格納するメモリ（202）と、前記メモリ（202）に接続され、前記複数のタスク記述ブロック（250）のそれぞれにアクセスし、前記タスク記述ブロック（250）で識別される情報に基づいて前記パリティ演算を実行するパリティ演算ロジック（205）と、を備えるディスクコントローラ。

【０１１１】２．前記メモリ（202）は、それぞれが前記複数のタスク記述ブロック（250）のうちの１つを識別する複数のエントリを有するタスク記述ブロック待ち行列（216）をさらに格納し、前記パリティ演算ロジック（205）は、前記複数のエントリのうちの１つによって識別されるタスク記述ブロック（250）にアクセスすることによって、パリティ演算の実行に必要な情報を得る、上記１に記載のディスクコントローラ（200）。

【０１１２】３．前記複数のタスク記述ブロック（250）はそれぞれ、対応するパリティ演算を実行するために必要な情報を格納し、前記情報には、前記パリティ演算に関するディスクデータについて、前記メモリ（202）内の位置をそれぞれ識別する複数のディスクデータポインタ（254）と、前記パリティ演算の結果を格納すべき前記メモリ（202）内の位置を識別する出力ポインタ（256）と、前記パリティ演算に関するパリティ係数が格納される前記メモリ（202）内の位置を識別する係数ポインタ（258）と、実行すべき前記パリティ演算のタイプの識別子（253）と、が含まれる、上記１に記載のディスクコントローラ。

【０１１３】４．パリティ演算についての情報をメモリ（202）に格納するステップ（272、２７４、276、280、282）であって、前記パリティ演算は、冗長データ記憶システムにおけるストライプについて少なくとも１つのパリティセグメントを生成するか、または前記ストライプについて少なくとも１つのデータセグメントを再生成するものである、ステップと、前記パリティ演算に必要な前記情報がすべて前記メモリ（202）に格納された後でのみ、パリティ演算ロジック（205）が前記情報を利用できるようにするステップと、を含む方法。

【０１１４】５．前記格納ステップは、前記パリティ演算についての前記情報を前記メモリ（202）に格納するステップと、前記情報を指すポインタを前記パリティ演算に対応するタスク記述ブロック（250）に格納するステップと、を含む上記４に記載の方法。

【０１１５】６．前記格納ステップは、前記情報のメモリ位置識別子を前記パリティ演算に対応するタスク記述ブロック（250）に格納するステップを含み、前記情報を利用できるようにするステップは、前記タスク記述ブロック（250）の識別子を、前記パリティ演算ロジック（205）がアクセス可能なタスク記述ブロック待ち行列（216）に追加するステップを含む、上記４に記載の方法。

【０１１６】７．前記格納ステップは、前記メモリ（202）内にタスク記述ブロック（250）を生成するステップ（270）を含み、該タスク記述ブロック（250）には、前記パリティ演算に関するディスクデータの、前記メモリ（202）内の位置をそれぞれ識別する複数のディスクデータポインタ（254）と、前記パリティ演算の結果を格納すべき前記メモリ（202）内の位置を識別する出力ポインタ（256）と、前記パリティ演算に関するパリティ係数が格納される前記メモリ（202）内の位置を識別す

る係数ポインタ（258）と、実行すべき前記パリティ演算のタイプの識別子（253）と、を含む、上記４に記載の方法。

【０１１７】８．ハードウェア演算ロジック（205）が情報を利用できるようにして、１つまたは複数のデータセグメントに基づいて演算を実行する方法であって、演算に関する１つまたは複数のデータセグメントをそれぞれメモリ（202）内の別々の位置に格納するステップと、前記演算に対応するタスク記述ブロック（250）に、前記メモリ（202）内の別々の位置それぞれへのポインタ（254）を格納するステップと、前記タスク記述ブロック（250）に、前記演算の結果を格納すべき前記メモリ（202）内の結果位置へのポインタ（256）を格納するステップと、タスク記述ブロック待ち行列（216）に、前記タスク記述ブロック（250）へのポインタ（252）を格納するステップと、を含む方法。

【０１１８】９．前記ハードウェア演算ロジック（205）が前記パリティ演算を実行し、前記方法は、前記メモリ（202）に異なる前記パリティ演算に使用される複数の係数を格納するステップと、前記タスク記述ブロック（250）に、前記パリティ演算ロジック（205）が前記パリティ演算に使用する複数の係数サブセットへのポインタ（258）を格納するステップと、をさらに含む、上記８に記載の方法。

【０１１９】１０．前記ポインタ（258）を前記タスク記述ブロック（250）に格納するステップは、前記メモリ（202）内の前記複数の異なる位置それぞれを指す前記ポインタ（254）が前記タスク記述ブロック（250）に格納された後でのみ、かつ前記結果位置を指す前記ポインタ（256）が前記タスク記述ブロック（250）に格納された後でのみ、前記タスク記述ブロック（250）を指す前記ポインタを前記タスク記述ブロック待ち行列（216）に格納するステップを含む、上記８に記載の方法。

【図面の簡単な説明】

【図１】従来技術によるＮ＋１冗長グループ化を示すブロック図である。

【図２】本発明の実施形態において使用されるＮ＋２冗長グループ化を示すブロック図である。

【図３】本発明によるメモリテーブルのレイアウトを示すブロック図である。

【図４】本発明の特定の実施形態によりパリティ演算を実行する方法を示す図である。

【図５】本発明によるディスクコントローラの関連コンポーネントを示すブロック図である。

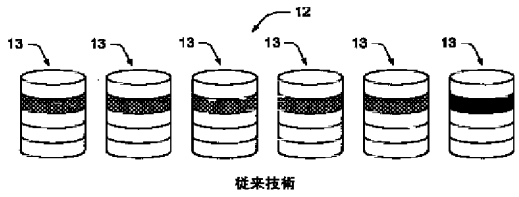
【図６】タスク記述ブロックの例示的な構造をさらに詳細に示す図である。

【図７】本発明の特定の実施形態によるタスク記述ブロックを使用する方法を示す図である。

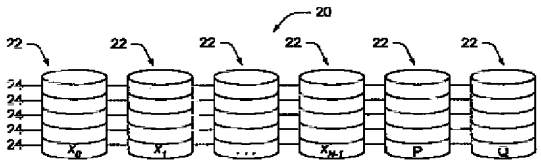
【符号の説明】

200　ディスクコントローラ
202　メモリ
205　演算ロジック
216　タスク記述ブロック待ち行列
250　タスク記述ブロック
252　タスク記述ブロックポインタ
253　ヘッダ
254　ディスクデータポインタ
256　出力ポインタ
258　係数ポインタ及びデータ

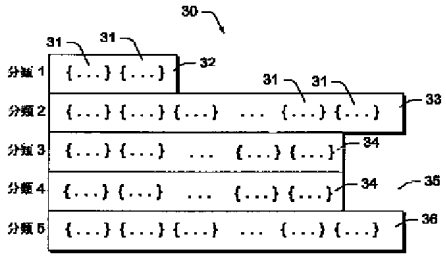
【図１】



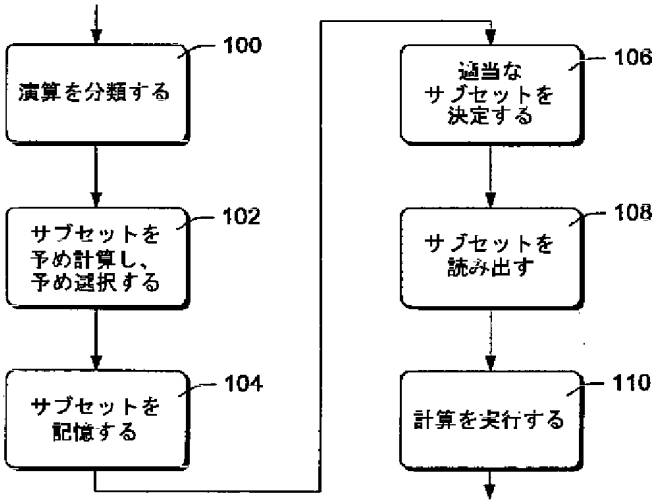
【図２】



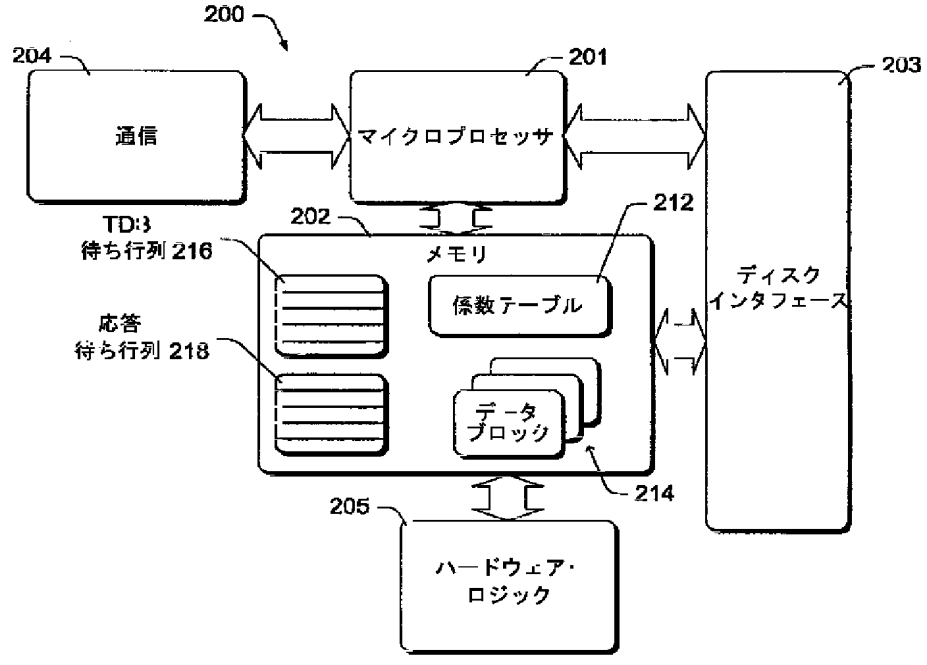
【図３】



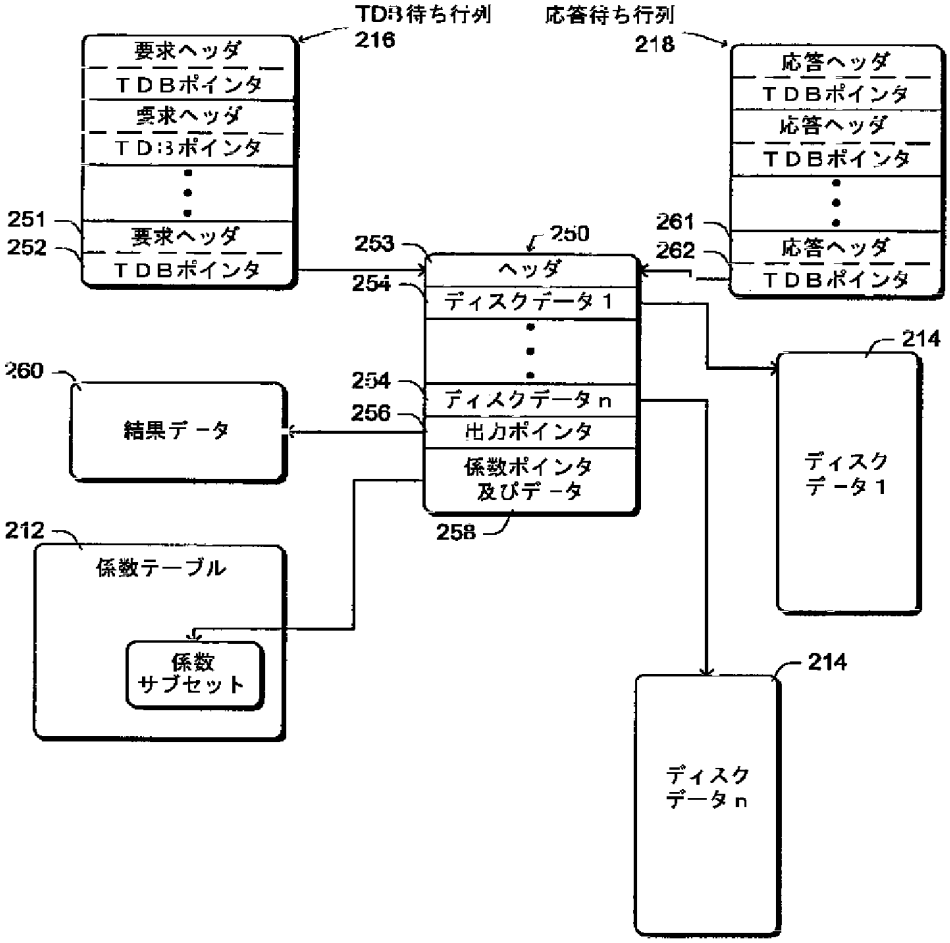
【図4】


100
演算を分類する
102
サブセットを
予め計算し、
予め選択する
104
サブセットを
記憶する
106
適当な
サブセットを
決定する
108
サブセットを
読み出す
110
計算を実行する

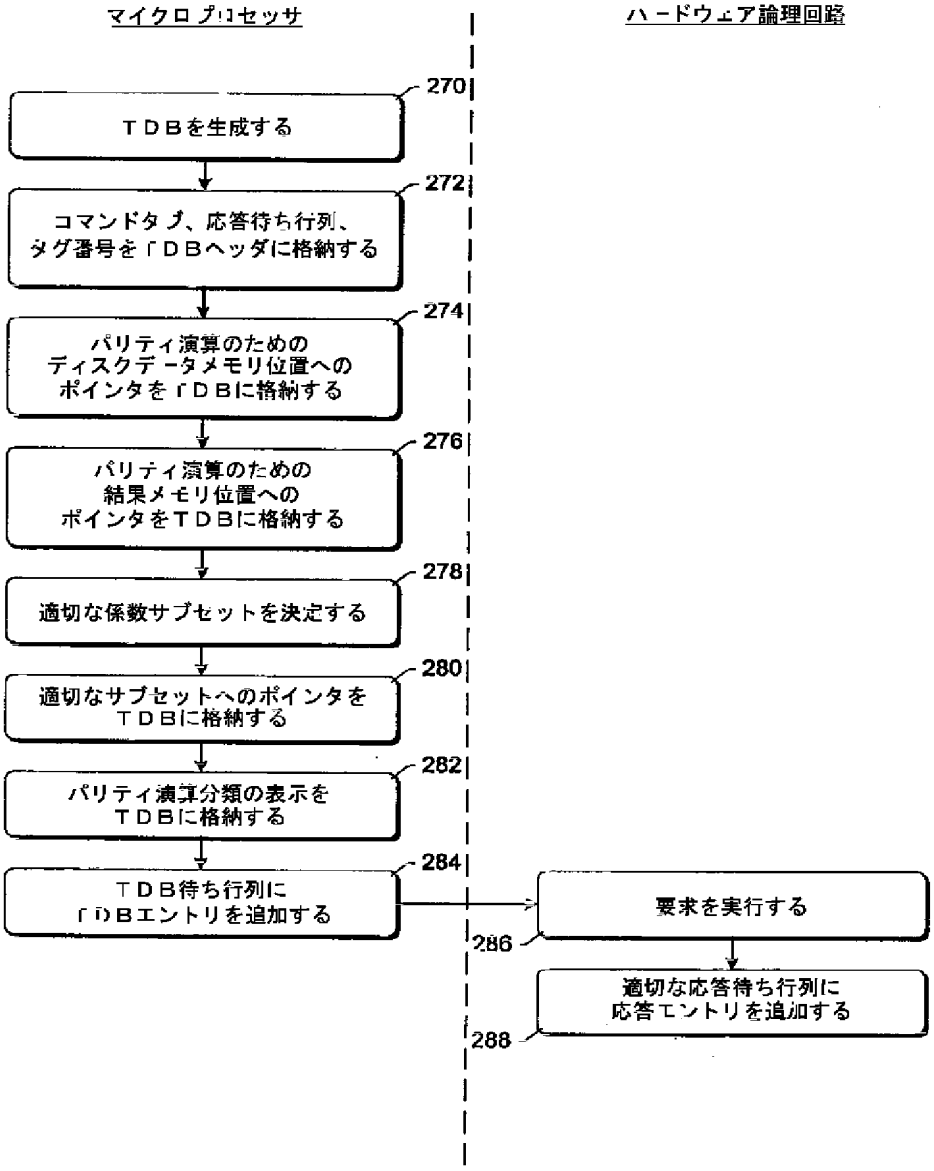
【図5】


200
204
通信
201
マイクロプロセッサ
203
ディスク
インタフェース
202
メモリ
212
TDB
待ち行列 216
応答
待ち行列 218
係数テーブル
データ
ブロック
214
205
ハードウェア・
ロジック

【図6】


TDB待ち行列
216
応答待ち行列
218
要求ヘッダ
TDBポインタ
要求ヘッダ
TDBポインタ
要求ヘッダ
TDBポインタ
251
252
253
250
ヘッダ
254
ディスクデータ1
ディスクデータn
256
出力ポインタ
係数ポインタ及びデータ
258
261
262
応答ヘッダ
TDBポインタ
応答ヘッダ
TDBポインタ
応答ヘッダ
TDBポインタ
260
結果データ
212
係数テーブル
係数サブセット
214
ディスクデータ1
214
ディスクデータn

【図7】



フロントページの続き

(72)発明者　ロバート・エー・ラスト
アメリカ合衆国83704アイダホ州ボイジー
アラマー・ドライブ　5061

Fターム(参考)　5B001 AA02 AB03 AC01 AD03 AE02
5B065 BA01 EA03